ONLINE LEARNING PARTIAL LEAST SQUARES REGRESSION MODEL FOR UNIVARIATE RESPONSE DATA

Lei Qin[*]*, Hichem Snoussi*[*]*, Fahed Abdallah*[†]

[*] Université de Technologie de Troyes
Institute Charles Delaunay
12 Rue Marie Curie
10004, Troyes, France

[†] Université de Technologie de Compiègne
Laboratoire Heudyasic
Rue Roger Couttolenc
60203, Compiègne, France

ABSTRACT

Partial least squares (PLS) analysis has attracted increasing attentions in image and video processing. Currently, most applications employ batch-form PLS methods, which require maintaining previous training data and re-training the model when new observations are available. In this work, we propose a novel approach that is able to update the PLS model in an online fashion. The proposed approach has the appealing property of constant computational complexity and const space complexity. Two extensions are proposed as well. First, we extend the method to be able to update the model when some training samples are removed. Second, we develop a weighted version, where different weights can be assigned to the data blocks when updating the model. Experiments on real image data confirmed the effectiveness of the proposed methods.

Index Terms— Partial Least Squares Analysis, image processing, online learning

1. INTRODUCTION

Partial Least Squares (PLS) regression is a statistical method which models relations between sets of observed variables X and Y by means of latent variables. It constructs new predictor variables, known as components, as linear combinations of the original predictor variables, with consideration of the observed response values. According to whether Y is a vector or a matrix, PLS is categorized into two algorithms, the PLS-1 and PLS-2. Furthermore, by setting Y as categorical labels, PLS can be applied as a discriminant tool for the estimation of a low dimensional space that maximizes the separation between samples of different classes. This is the so-called Partial least squares Discriminant Analysis (PLS-DA).

Recently, PLS-DA has attracted increasing attentions in image/video processing and computer vision. It has been successfully applied to pedestrian detection [1], face identification [2, 3], discriminative appearance model learning [4] and object tracking [5]. Despite its increasing popularity, all the aforementioned applications employed batch PLS algorithms, e.g. NIPALS [6] or SIMPLS [7], which require maintaining all the training samples and retrain the PLS model each time when some new training data are available. Due to their storage and computational requirements, these batch methods are unsatisfactory for real-world applications. First, they use the entire set of training samples for each update. If an update is made at each time step, then the number of samples which must be retained grows linearly with the length of the time series. Second, the cost of computation grows with the number of samples, so they will run ever slower as time progresses.

To the best of our knowledge, few approaches have been proposed in the literature for incrementally updating a PLS model. This may be due to the iterative nature of the classic PLS algorithms, e.g. NIPALS and SIMPLS, which makes incremental methods not straightforward. In this work, we limit our discussion to the PLS analysis problems with a single response variable, i.e. the PLS-1 algorithm, and propose online PLS-1 algorithms that can update a PLS-1 model without retraining. A weighted extension is proposed as well which enables the updating method to assign weights to different training data blocks. Analysis reveals that the proposed online updating algorithms possess the appealing property of constant storage and computational complexities while being accurate compared to their batch counterparts.

The remainder of this paper is organized as follows: we briefly review PLS in Section 2 and introduce a closed-form PLS-1 solution in section 2.1. The incremental and decremental PLS-1 model updating methods are presented in Section 3. A weighted extension is addressed as well. Experiments on real-world image data are shown in Section 4. We conclude in Section 5.

2. THE PLS ANALYSIS

Let $X \in \Re^{N \times r}$ be a mean-centered matrix of predictor variables, with rows corresponding to observations and columns to variables and $Y \in \Re^{N \times m}$ be the mean-centered response matrix. PLS methods find new spaces where most variations of the observed samples can be preserved, and the learned la-

tent variables from two blocks are more correlated than those in the original spaces

$$\begin{aligned} X &= TP^\top + E \\ Y &= UQ^\top + F \end{aligned} \tag{1}$$

where $T \in \Re^{N\times p}$ and $U \in \Re^{N\times p}$ are factor (score, component, latent variable) matrices, $P \in \Re^{r\times p}$ and $Q \in \Re^{m\times p}$ are loading matrices, and $E \in \Re^{N\times r}$ and $F \in \Re^{N\times m}$ are error terms. Discriminative features T are extracted and the dimension is reduced when $p < r$.

To decompose X and Y by T and U, an intermediate weighting matrix W is usually employed. For space reason, we refer the readers to [6] and [8] for the details of the classical NIPALS procedure for computing the PLS model. After training, the overall regression coefficient β is learned and stored for testing new samples. Specifically, for a test feature vector x_t, its regression response y_t is evaluated by

$$y_t = (x_t - \mu(X))^\top \beta + \mu(Y), \tag{2}$$

where $\mu(X)$ and $\mu(Y)$ are the sample means of X and Y before the mean-centering respectively. For additional details about batch PLS methods, we refer the readers to [9].

2.1. A Closed-Form PLS-1 Solution

The classical PLS algorithms, e.g. NIPALS [6] and SIMPLS [7], are iterative procedures. In addition, the usage of the raw data blocks X and Y are required at each iteration. The iterative nature and the dependency on the raw data make it difficult to develop online algorithms basing on these methods.

Alternatively, in [10], a closed-form PLS-1 solution is proposed. It takes two scatter matrices, namely S_{xx} and S_{xy}, as input to compute the PLS model instead of using the raw data block X and Y. The two scatter matrices are defined as

$$S_{xx} = \sum_{i=1}^{N} (x_i - \mu(X))(x_i - \mu(X))^\top \tag{3}$$

$$S_{xy} = \sum_{i=1}^{N} (x_i - \mu(X))(y_i - \mu(Y))^\top, \tag{4}$$

where N is the number of samples in X (and also in Y) and $\mu(X)$ and $\mu(Y)$ are sample means of X and Y respectively. Note that in (3) and (4), each x_i and y_i are arranged in vector form and we have $S_{xx} \in \Re^{r\times r}$ and $S_{xy} \in \Re^{r\times m}$.

Using S_{xx} and S_{xy}, the Krylov Matrix $K_r \in \Re^{r\times rm}$ of the pair (S_{xx}, S_{xy}) is defined as

$$K_r = [S_{xy} \quad S_{xx}S_{xy} \quad S_{xx}^2 S_{xy} \quad \cdots \quad S_{xx}^{r-1} S_{xy}]. \tag{5}$$

A reduced Krylov matrix $K_p \in \Re^{r\times pm}$ is formed by the first p $(1 \le p \le r)$ columns of K_r:

$$K_p = [S_{xy} \quad S_{xx}S_{xy} \quad S_{xx}^2 S_{xy} \quad \cdots \quad S_{xx}^{p-1} S_{xy}]. \tag{6}$$

Further, the relationship between the weight matrix W of the trained PLS model and the Krylov matrix of the pair (S_{xx}, S_{xy}) is recognized. It is revealed that for univariate Y, i.e. when $m = 1$, the conventional orthonormal weighting matrix $W_p \in \Re^{r\times p}$ using p latent variables and the Krylov matrix K_p span the same column space. Moreover, W_p can be computed directly by performing the QR decomposition (and take the Q part) or the (modified) Gram-Schmidt procedure on K_p.

Furthermore, the regression coefficient β can be computed in a direct formula either using K_p as

$$\beta_{K_p} = K_p (K_p^\top S_{xx} K_p)^{-1} K_p^\top S_{xy} \tag{7}$$

or using W_p as

$$\beta_{W_p} = W_p (W_a^\top S_{xx} W_p)^{-1} W_p^\top S_{xy}. \tag{8}$$

The two formulations β_{K_p} and β_{W_p} yield identical results because K_p and W_p span the same column space [10]. They correspond to the Partial Least Squares (PLS) regression using p latent variables when $p < r$ and reduce to the Ordinary Least Squares (OLS) regression (assuming that S_{xx} is non-singular) when $p = r$.

In practice, the explicitly formulated Krylov matrix K_p in Equation (6) may be ill-conditioned due to accumulated round off errors when computing the powers of S_{xx}, especially when p is large. This adversely affects the accuracy of the resulting W_p and β. As suggested in [10], we use the Arnoldi's method [11] to extract the orthonormal basis of K_p from S_{xx} and S_{xy}. The pseudo code procedure of the Arnoldi's method for computing W_p is described in Algorithm 1, where $\|\cdot\|_F$ is the Frobenius norm. The regression coefficient β can thus be obtained using the resulting W_p according to Equation (8). For detailed proof and further information of the closed-form PLS-1 solution, we refer the readers to [10].

Algorithm 1 Arnoldi's method for computing orthonormal weight matrix W_p

Require: S_{xx} and S_{xy}: the scatter matrices
p: the number of retained components
Ensure: the weight matrix W_p
1: $w_1 \leftarrow S_{xy}/\|S_{xy}\|_F$
2: **for** $i = 2 \cdots p$ **do**
3: $w_i \leftarrow S_{xx} w_{i-1}$
4: **for** $j = 1 \cdots i-1$ **do**
5: $h_{j,i-1} \leftarrow w_j^\top w_i$
6: $w_i \leftarrow w_i - h_{j,i-1} w_j$
7: **end for**
8: $h_{i,i-1} \leftarrow \|w_i\|_F$
9: $w_i \leftarrow \frac{w_i}{h_{i,i-1}}$
10: **end for**
11: $W_p = [w_1\ w_2\ \cdots\ w_p]$

3. ONLINE PLS-1 MODEL UPDATING METHODS

It is worth noting that the two scatter matrices S_{xx} and S_{xy} are constant in size, i.e. independent of N, and can be updated incrementally with new samples. An incremental PLS model updating algorithm can thus be derived. In deed, the output W and β of a PLS model trained from the data blocks X and Y can be fully determined by S_{xx} , S_{xy} and the dimension of retained latent variables p using Algorithm 1 and Equation (8) respectively. Besides, in order to update S_{xx} and S_{xy}, it is necessary to store the number of samples, $N(X)$, and the samples means, $\mu(X)$ and $\mu(Y)$. Therefore, we suggest to specify a PLS model trained from the data block X and Y as

$$\Theta(X, Y, p) = (N(X), \mu(X), \mu(Y), S_{xx}, S_{xy}, W, \beta). \quad (9)$$

The advantage of adopting this model is that all the elements specified in the model are of constant size (independent of the number of training samples), making the model have constant space complexity.

3.1. Incremental Model Updating

Suppose we have trained a PLS model using training set X_1 and Y_1 with dimension p_1. The model is then denoted as $\Theta(X_1, Y_1, p_1)$. When some new samples, e.g. feature vectors X_2 and their corresponding labels Y_2, are available, the incremental updating algorithm seeks to update the PLS model Θ with X_2 and Y_2 without resorting to the original training set X_1 and Y_1.

Each element in the model is updated as follows. Firstly, the first five elements for $\Theta(X_2, Y_2, p_2)$ is computed as $N(X_2)$, $\mu(X_2)$, $\mu(Y_2)$, S_{xx2}, S_{xy2} respectively. Incremental updating of $N(X)$, $\mu(X)$ and $\mu(Y)$ is straightforward:

$$N(X) = N(X_1) + N(X_2); \quad (10)$$

$$\mu(X) = \frac{N(X_1)}{N(X)}\mu(X_1) + \frac{N(X_2)}{N(X)}\mu(X_2), \quad (11)$$

$$\mu(Y) = \frac{N(X_1)}{N(X)}\mu(Y_1) + \frac{N(X_2)}{N(X)}\mu(Y_2). \quad (12)$$

The scatter matrix S_{xx} can be updated using the following equation:

$$S_{xx} = S_{xx1} + S_{xx2} + \frac{N(X_1)N(X_2)}{N(X)}(\mu(X_1) - \mu(X_2))(\mu(X_1) - \mu(X_2))^\top. \quad (13)$$

Similarly, S_{xy} can also be updated as

$$S_{xy} = S_{xy1} + S_{xy2} + \frac{N(X_1)N(X_2)}{N(X)}(\mu(X_1) - \mu(X_2))(\mu(Y_1) - \mu(Y_2))^\top. \quad (14)$$

The weight matrix W can thus be updated using the newly updated S_{xx} and S_{xy} according to Algorithm 1. Finally, the regression coefficient β is updated by Equation (8). We note that although p can be different from both p_1 and p_2, there is no loss of information because the number of samples, the means and the scatter matrices have encoded all the information needed to update a PLS-1 model.

3.2. Decremental Model Updating

It is interesting to note that in some applications, one seeks to adjust the model after removing some trained samples. Now we have trained a PLS model on the dataset of X_1 and Y_1, we need to update the model after removing a training data block X_2 as well as its corresponding response Y_2. This is the decremental PLS (DPLS) model updating problem.

This is a straightforward extension of the incremental updating procedure. We update the number of data, and their means:

$$N(X) = N(X_1) - N(X_2) \quad (15)$$

$$\mu(X) = \frac{N(X_1)}{N(X)}\mu(X_1) - \frac{N(X_2)}{N(X)}\mu(X_2), \quad (16)$$

$$\mu(Y) = \frac{N(X_1)}{N(X)}\mu(Y_1) - \frac{N(X_2)}{N(X)}\mu(Y_2). \quad (17)$$

Then it is not difficult to prove that the scatter matrices S_{xx}, S_{xy} can be updated as

$$S_{xx} = S_{xx1} - S_{xx2} - \frac{N(X_1)N(X_2)}{N(X)}(\mu(X_1) - \mu(X_2))(\mu(X_1) - \mu(X_2))^\top. \quad (18)$$

Similarly, S_{xy} can also be updated as

$$S_{xy} = S_{xy1} - S_{xy2} - \frac{N(X_1)N(X_2)}{N(X)}(\mu(X_1) - \mu(X_2))(\mu(Y_1) - \mu(Y_2))^\top. \quad (19)$$

The weight matrix W and the regression coefficient β are then updated using Algorithm 1 and Equation (8) respectively using the newly updated S_{xx} and S_{xy}.

3.3. Weighted Model Updating

In some applications, it is interesting to give different weights to different training samples when updating the model. For example, in visual tracking, when the target undergoes the appearance changes, it is likely that the recent observations will be more indicative of its appearance than the more ancient ones. Therefore, it may be desirable to focus more on recently-acquired images and down-weight the contribution of earlier observations. On the contrary, for semi-supervised learning, a classifier is trained using labeled data, it exploits a set of unlabeled data to improve its accuracy. In this case, one may need to give smaller weights to the unlabeled samples.

To tackle this problem, we propose a weighted extension of the IPLS called weighted incremental PLS (WIPLS) model

updating method. The key idea is the concept of the "effective number" of a sample. By default, all observations have the same weight of 1.0. If a sample is assigned with a weight of 2.0, the result would be the same as if we had repeated this sample twice when counting the sample number, computing the means and the scatter matrices. On the other extreme, a point associated with a weight of 0 would make the result as if it had not been included in the computation at all.

For WIPLS, we assign weights to the two training blocks with two scalar factors f_1 and f_2 when updating the model. The effective number of samples $N(X)$ and sample means $\mu(X), \mu(Y)$ are updated with the weight factor f_1 and f_2 as

$$N(X) = f_1 N(X_1) + f_2 N(X_2), \tag{20}$$

$$\mu(X) = \frac{f_1 N(X_1)}{N(X)}\mu(X_1) + \frac{f_2 N(X_2)}{N(X)}\mu(X_2), \tag{21}$$

$$\mu(Y) = \frac{f_1 N(X_1)}{N(X)}\mu(Y_1) + \frac{f_2 N(X_2)}{N(X)}\mu(Y_2). \tag{22}$$

The scatter matrix S_{xx} can be updated using the following equation:

$$S_{xx} = f_1 S_{xx1} + f_2 S_{xx2} + \frac{f_1 f_2 N(X_1) N(X_2)}{N(X)}(\mu(X_1) - \mu(X_2))(\mu(X_1) - \mu(X_2))^\top. \tag{23}$$

Similarly, S_{xy} is updated with forgetting factor f as

$$S_{xy} = f_1 S_{xy1} + f_2 S_{xy2} + \frac{f_1 f_2 N(X_1) N(X_2)}{N(X)}(\mu(X_1) - \mu(X_2))(\mu(Y_1) - \mu(Y_2))^\top. \tag{24}$$

Finally, the regression model W and β can be updated via Algorithm 1 and Equation (8) respectively using the newly updated S_{xx} and S_{xy}.

It is easy to observe that when $f_1 = f_2 = 1.0$, WIPLS is identical to IPLS. Similarly, WIPLS reduces to DPLS when $f_1 = 1.0$ and $f_2 = -1.0$. This indicates that WIPSL is the general method for updating PLS model and both IPLS and DPIS are special cases of WIPLS. Besides, it is worth noting that when $0 < f_1 < 1.0$ and $f_2 = 1.0$, f_1 is the so-called the "forgetting factor" because it weights less (forgets) the previously trained samples.

4. EXPERIMENTS

In order to validate the effectiveness of the incremental and decremental PLS model updating approaches proposed in the previous section, we conducted an empirical study on benchmark data set from UCI Repository [12]. The used "Relative Location of CT Slices on Axial Axis" data set consists of 384 features extracted from 53500 CT images from 74 different patients. The class variable is numeric and denotes the relative location of the CT slice on the axial axis of the human body.

We compared IPLS and DPLS with their batch counterparts. Without confusion, we denote PLS as the batch PLS methods in this section. For PLS, we employed the two most popular algorithms, NIPALS [6] and SIMPLS [7]. The following strategy was taken: the training sample was provided in an online way, with 100 new samples at each following step. At the initial step, both the PLS methods and the IPLS approach trained a model using the initial 100 samples respectively. When new samples were available, PLS methods had to retrain the model and IPLS could update the model online according to the procedure described in Section 3.1. As there were 53500 samples in total, PLS retrained the model 534 times and IPLS updated the model also 534 times. The number of retained latent variables P was set to 15 for all the three methods.

The experiments were carried out by running Matlab implementations on a desktop with 2.30GHz CPU and 12 GB memory. We recoded the Frobenieus norm of difference of the three weight matrices W and the three regression coefficients β, that were computed by the three methods at each step. Concerning the weight matrix of SIMPLS, it is not the same as those produced by NIPALS or IPLS. However, it shares the same column space with the other two weight matrices. Since W of NIPALS and IPLS are orthonormal, we first performed a QR decomposition of the raw weight matrix of SIMPLS and then took the orthogonal basis Q for comparison. In addition to the norm of difference, the computational time for (re)training or updating the models at each step was recorded as well.

Figure 1 shows the computation time of the three methods each time when they retrain or update their models respectively. Not surprisingly, computational time of both NIPALS and SIMPLS grew linearly with the number of training samples. For NIPALS, average computational time was 1.1145 seconds and the value for SIMPLS was 0.1938 seconds. In contrast, computational time for IPLS was almost constant and average processing time was 0.0057 seconds.

In terms of accuracy, the average norm of difference between the weight matrices W produced by IPLS and that of NIPAL or SIMPLS (we took a maximum) during the 534 updates was $4.8131e^{-012}$, with a maximum value of $4.2417e^{-011}$. Likewise, the norm of difference between the regression coefficients β had an average value of $6.4392e^{-012}$ and a maximum value of $1.7628e^{-011}$.

For evaluating the DPLS method, we re-ran the experiments in a reverse way. We began with 53500 samples in the initial step and removed 100 samples at each step. Consequently, there were 534 times of retraining for NIPALS and SIMPLS and 534 times of updating for DPLS. For DPLS, the initial model was taken from the the final model produced by IPLS in the last experiment.

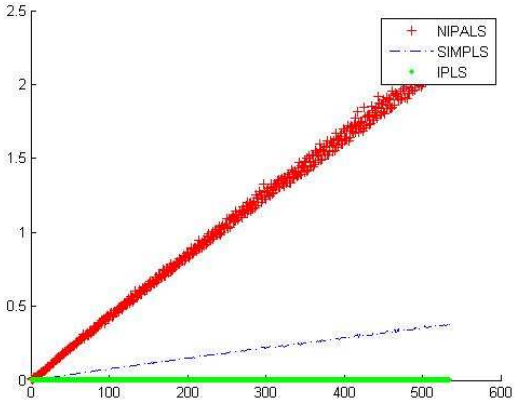

Fig. 1. Computational time for NIPALS, SIMPLS and IPLS. The horizontal axis is the experimental steps (535 in total) and the vertical axis is the computational time in seconds.

As expected, our results showed that the computational time of NIPLS and SIMPLS in this setting decreased linearly with the number of training samples. Average processing time for NIPALS was 1.1053 seconds. It was 0.1918 seconds for SIMPLS and 0.0056 seconds for DPLS. The average norm of difference of W between DPLS and NIPALS or SIMPLS (the larger one was taken) was $1.2621e^{-009}$. The maximum norm of difference was $5.3754e^{-007}$. Average norm of difference of β was $7.2808e^{-010}$ with a maximum value of $2.1860e^{-007}$.

We see from the above results that the proposed IPLS and DPLS methods are both accurate and efficient. In terms of accuracy, the differences is still negligible after thousands of times of updating. On the other hand, substantial time gain is achieved using IPLS or DPLS. Although we didn't explicitly measure the space complexity, it is easy to see that the proposed IPLS and DPLS methods have constant space complexity.

It may be arguable whether our methods are genuinely online PLS methods because they do not update the W and β directly. Instead, they update some intermediate representations, i.e. the scatter matrices. Nevertheless, our methods are of practical use because they are accurate and have constant time complexity as demonstrated in the experiments.

5. CONCLUSIONS

The proposed online PLS-1 methods have constant time and space complexities. The incremental and the decremental model updating methods are special cases of a generalized weighted extension, which can assign weights to different training data blocks when updating the model. Experiments demonstrate that the proposed methods are both accurate and efficient. We thus believe that the proposed online PLS-1 methods can be of interest to researchers in image and video processing, computer vision, and related fields.

REFERENCES

[1] W. R. Schwartz, A. Kembhavi, D. Harwood, and L. S. Davis, "Human detection using partial least squares analysis," in *IEEE International Conference on Computer Vision*, 2009, pp. 24–31.

[2] W. R. Schwartz, H. Guo, J. Choi, and L. S. Davis, "Face identification using large feature sets," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2245–2255, 2012.

[3] G. Chiachia, N. Pinto, W. R. Schwartz, A. Rocha, A. X. Falcao, and D. Cox, "Person-specific subspace analysis for unconstrained familiar face identification," in *British Machine Vision Conference*, 2012.

[4] W. R. Schwartz and L. S. Davis, "Learning Discriminative Appearance-Based Models Using Partial Least Squares," in *Brazilian Symposium on Computer Graphics and Image Processing*, 2009.

[5] Lei Qin, Hichem Snoussi, and Fahed Abdallah, "Cascaded generative and discriminative learning for visual tracking," in *Image Analysis and Recognitio*, Mohamed Kamel and Aurélio Campilho, Eds. 2013, vol. 7950 of *Lecture Notes in Computer Science*, pp. 397–406, Springer.

[6] Herman Wold, "Path models with latent variables: The nipals approach," in *Quantitative Sociology: International perspectives on mathematical and statistical modeling*, H M Blalock, A Aganbegian, F M Borodkin, R Boudon, and V Capecchi, Eds. 1975, pp. 307–357, Academic Press.

[7] Sijmen de Jong, "Simpls: An alternative approach to partial least squares regression," *Chemometrics and Intelligent Laboratory Systems*, vol. 18, pp. 251–263, March 1993.

[8] Inge Helland, *Partial Least Squares Regression*, vol. 6, Wiley, New York, 1985.

[9] Roman Rosipal and Nicole Krämer, "Overview and recent advances in partial least squares," in *Lecture Notes in Computer Science*, 2006, vol. 3940, pp. 34–51.

[10] David di Ruscio, "A weighted view on the partial least-squares algorithm," *Automatica*, vol. 36, pp. 831–850, 2000.

[11] Walter Edwin Arnoldi, "The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Quarterly of Applied Mathematics*, vol. 9, no. 17, pp. 17–29, 1951.

[12] K. Bache and M. Lichman, "UCI machine learning repository," 2013.